Application for United States Letters Patent

for

Compression and Synthesis of Two Dimensional Images

by

Michael H. Chu

Radek Grzeszczuk

Jean-Yves Bouguet

Prepared by:

Seth Z. Kalson
Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95052-8119

"Express Mail" mailing label number: EL672754112US
Date of Deposit: 6/28/01
I hereby certify that I am causing this paper or fee to be deposited with the United States Postal Service "Express Mail Post Office to Addressee" service on the date indicated above and that this paper or fee has been addressed to the Commissioner for Patents, Washington, D. C. 20231

D. Roberts
(Typed or printed name of person mailing paper or fee)

D. Roberts
(Signature of person mailing paper or fee)

6/28/01
(Date signed)

Field

[0001] Embodiments of the present invention relate to image processing, and more particularly, to image compression and synthesis.

Background

[0002] Digital communications is an important feature of personal computers. Data traffic carried by a digital communication system often includes image data, so that image rendering is an important task carried out by a computer system. Fig. 1 is a high level depiction of a portion of a computer system, where microprocessor **102** performs various computational tasks under control of one or more programs stored in memory **104**. Memory traffic is handled by chipset **106**, which includes a memory controller connected to memory bus **108**. Microprocessor **102** communicates with chipset **106** via front side bus **110**. Chipset **106** may also include a functional unit for communicating with graphics hardware **112** via graphics port **114**.

[0003] Chipset **106** also allows microprocessor **102** to communicate with other peripheral components, such as network interface **114** via system bus **116**. Network interface **114** allows the computer system to communicate with other nodes on a network, such as servers, gateways, etc., for receiving various forms of data traffic, such as image data. Alternatively, a modem (not shown) may be connected to system bus **116**, allowing communication to the public switched telephone system, whereby communication may be made to other networks, including the Internet.

[0004] Fig. 1 is meant to serve only has a high level description of only one embodiment of a computer system. Other embodiments may have graphics hardware connected to system bus **116**, or may have graphics hardware embedded in microprocessor **102**.

[0005] Image data is very bandwidth intensive, and often is compressed (encoded) before sending it to another node on a network. For example, JPEG (Joint Photographic Experts Group) provides standards, such as ITU-TT.81, for encoding and decoding images. A compressed image received by network interface **114** is eventually uncompressed (decompressed, decoded, synthesized, or reconstructed) by either microprocessor **102** or graphics hardware **114**, or perhaps another graphics accelerator in the computer system.

[0006] Graphics hardware **112** may be dedicated for quickly rendering image data to a monitor (not shown). To support the fast rendering of images, graphics hardware **112** may be designed to support various forms of vector processing in a very efficient manner. For example, multiplication of a one dimensional vector by a scalar may be performed very quickly for some high performance graphics hardware. It is desirable to provide a compression and synthesis scheme that provides a good compression ratio and for which image synthesis takes advantage of vector processing.

Brief Description of the Drawings

[0007] Fig. 1 is a high-level abstraction of a computer system

[0008] Fig. 2 is a simplified example of image matrix partitioning according to an embodiment of the present invention.

[0009] Fig. 3 is a flow diagram for image compression according to an embodiment of the present invention.

[0010] Fig. 4 is a flow diagram for image synthesis according to an embodiment of the present invention.

[0011] Fig. 5 is a flow diagram for image synthesis according to another embodiment of the present invention.

Description of Embodiments

[0012] A two dimensional image may be represented as a matrix M, not necessarily square, in which the image pixel values are the matrix elements $M(i,j)$, where i is the row index and j is the column index. There may be separate matrices for each color component of an image, such as the red, green, and blue color components. The problem at hand is to compress the image M, that is, to represent the image by some set of bits fewer in number than that needed for exact representation, and then to synthesize the image to obtain an image $\hat{M}$, where a goal may be for $\hat{M}$ to be perceived by an observer to be a close approximation to the original image M.

[0013] For simplicity of discussion, the embodiments of the present invention will be described for a single color component of an image. The compression and synthesis algorithms described herein may be applied to each color component of the image separately, so that M is an array of pixels for a particular color component.

[0014] Embodiments of the present invention divide an image M into sub-blocks in order to perform compression. Compression is performed on each sub-block. However, the sub-blocks are overlapping. A weight matrix is applied to each sub-block, and the weighted sub-blocks are represented by a weighted sum of vector outer products. The scalar weights and vectors used in the representation provide the parameters for compression of the image. The scalar weights and vectors are either stored, or communicated over a channel, whereupon synthesis of the approximate image $\hat{M}$ makes use of forming weighted vector outer products, and taking appropriate sums of these outer products.

[0015] Before describing the compression method in a more general and detailed manner, it is pedagogically useful to first provide a simplified example of how an image matrix may be partitioned into sub-blocks. In Fig. 2, pixel positions in an image M are indicated by points, where a set of points have been designated as vertices with index labels $i = 1, 2, \cdots, 25$. Let m_i for each $i = 1, 2, \cdots, 25$ denote a sub-block of pixels uniquely associated with each vertex i, defined as follows. For each vertex i, let $(J(i), K(i))$ denote the pixel coordinates for vertex i. The sub-blocks m_i are those matrix elements $M(j,k)$ for which $J(i) \le j \le J(i)+6$ and $K(i) \le k \le K(i)+6$.

[0016] For example, the boundaries of sub-block m_1 associated with vertex 1 are in bold, with its interior region cross-hatched. It is to be understood that the ranges for j and k do not exceed the ranges for the row and column indices, respectively, of M. For example, sub-block m_{19} associated with vertex 19 consists of the pixels within the square region defined by vertices 19, 20, 24, and 25. Note that sub-block m_{25} consists of only vertex 25.

[0017] Sub-block m_2 associated with vertex 2 is cross-hatched in a different direction from that of sub-block m_1. Note that these two sub-blocks overlap. In the particular embodiment of Fig. 2, at most only six sub-blocks have a non-zero intersection. For example, the intersection of sub-blocks $m_1, m_2, m_3, m_6, m_7, m_8$ is the solid line extending from vertex 8 to vertex 13. Fewer than six sub-blocks may have a larger

intersection. For example, the intersection of sub-blocks m_1, m_2, m_6, m_7 is the set of pixel elements in the square region defined by the four vertices 7, 8, 12, and 13.

[0018] To generalize to other embodiments, a set of V vertices in a matrix M are chosen, and labeled by an index i, where $i = 1, \cdots, V$. For each vertex i, a sub-block m_i may be defined as consisting of those matrix elements $M(j,k)$ for which $J(i) \le j \le J(i) + L_r(i)$ and $K(i) \le k \le K(i) + L_c(i)$, where $L_r(i)$ and $L_c(i)$ are integers such that $L_r(i)+1$ is the maximum number of vertices in the "row" direction of sub-block m_i and $L_c(i)+1$ is the maximum number of vertices in the "column" direction of m_i. For example, in Fig. 2 we have $L_r(i) = L_c(i) = 6, \forall i$.

[0019] Note that the example of Fig. 2 has the property that the corners of each sub-block lie at a vertex position. (For sub-blocks consisting of a line of pixel coordinates, such as for example sub-block m_{23} consisting of the pixel coordinates in the line extending from vertex 23 to vertex 25, the "corners" may be considered the endpoints of the line, and for the special case of sub-block m_{25}, the "corners" may be considered vertex 25.) In general, however, other embodiments may not have this property where sub-blocks are aligned on vertices, so that there need not be any particular relationship between $L_r(i)$, $L_c(i)$, and the coordinates of the vertices.

[0020] A matrix A may be decomposed into a sum of weighted outer products of one-dimensional vectors, given by

$$A = \sum_{i=1}^{N} \sigma_i u_i v_i' ,$$

where vectors u_i and v_i are column vectors, v_i' denotes the transpose of v_i, and σ_i is a scalar weight. For a real matrix A, the scalar weights may be ordered as $\sigma_1 \ge \sigma_2 \ge \cdots \ge \sigma_N$, and we assume that such an ordering is performed.

[0021] A well-known decomposition of this form is the singular value decomposition, where in this case $\sigma_i, i = 1, \cdots, N$ are the singular values. Singular value decomposition plays an important role in least squares problems. There are weighted outer product representations other than the singular value decomposition. However, for

the singular value decomposition, the sets of vectors $\{u_i, i=1,\cdots,N\}$ and $\{v_i, i=1,\cdots,N\}$ are orthonormal sets, and the singular value decomposition satisfies a least squares criterion.

[0022] For an embodiment of the present invention, each sub-block m_k is weighted by a weight matrix w_k to form a weighted sub-block $m_k * w_k$, where w_k has the same dimension as m_k and $*$ denotes element-by-element multiplication. The matrix weighting may not be uniform over the elements of any particular sub-block, nor perhaps may the weighting be uniform from sub-block to sub-block.

[0023] The weighted sub-block $m_k * w_k$ will have a weighted vector outer product sum decomposition

$$m_k * w_k = \sum_{i=1}^{N(k)} \sigma_i(k) u_i(k) v_i'(k).$$

As discussed above, this decomposition may be a singular value decomposition. A compression scheme is to represent each weighted sub-block by a subset of its associated scalar weights and vectors used in its outer sum decomposition. For some sub-blocks, the subset may not be a proper subset. However, compression is obtained by choosing these subsets to be proper subsets for most weighted sub-blocks.

[0024] For example, weighted sub-block $m_k * w_k$ may be represented by the set of scalar weights $\{\sigma_i(k), i=1,\cdots,n(k)\}$, the set of vectors $\{u_i(k), i=1,\cdots,n(k)\}$, and the set of vectors $\{v_i(k), i=1,\cdots,n(k)\}$, where $n(k) \le N(k)$. Various schemes may be used to choose $n(k)$. For one embodiment, $n(k)$ may be a chosen to be the lesser of $N(k)$ or some value C independent of k. For another embodiment, $n(k)$ may be a function of the size of m_k. For another embodiment, $n(k)$ may be chosen to be the smallest integer i such that $\sigma_{i+1}(k) < C$, where it is assumed that $\sigma_1(k) \ge \sigma_2(k) \ge \cdots \ge \sigma_{N(k)}(k)$ and C is independent of k, and if there is no such smallest integer, then $n(k) = N(k)$. Clearly, there are many schemes for choosing $n(k)$.

[0025] As a result, regardless of the particular method for choosing the subsets of scalar weights and associated vectors, the original matrix $\hat{M}$ is compressed into the family of sets of scalar weights

$$\{\{\sigma_i(k), i = 1, \cdots, n(k)\}, k = 1, \cdots, V\},$$

the family of sets of vectors

$$\{\{u_i(k), i = 1, \cdots, n(k)\}, k = 1, \cdots, V\},$$

and the family of sets of vectors

$$\{\{v_i(k), i = 1, \cdots, n(k)\}, k = 1, \cdots, V\}.$$

[0026] It should be noted that solving for a singular value decomposition is a non-linear problem, and in practice the singular values and associated vectors are computed to a some desired level of accuracy. Furthermore, after computation, the singular values and associated vectors may be quantized to some desired level of quantization for purposes of storage or digital communication. Therefore, it is to be understood in these letters patent, and in the claims herein, that reference to singular values and their associated vectors is to be interpreted to mean the singular values and associated vectors as represented in the finite arithmetic of the computer system or communication system over which the values are communicated. Similar comments apply to decompositions other than singular value decomposition, so that reference to scalar weights is to be interpreted to mean the scalar weights as represented in the finite arithmetic of the appropriate system.

[0027] A flow diagram for image compression according to an embodiment of the present invention is illustrated in Fig. 3. Starting with an image matrix M, in block **302** the image matrix M is partitioned into sub-blocks m_k. In block **304**, each sub-block m_k is weighed by a weight matrix w_k. In step **306**, a singular value decomposition is performed, or if performed iteratively, a partial singular value decomposition is performed, where only $n(k)$ singular values and associated vectors are retained for storage or communication. These singular values and associated vectors represent the compressed image. These values may be further quantized and perhaps encoded before storage or transmission over a communication channel.

[0028] Because the sub-blocks overlap, a particular image element will be in more than one sub-block and will be multiplied by different weight matrix elements

during different iterations of the flow diagram of Fig. 3. For example, in Fig. 2 consider the pixel element labeled p. Its pixel coordinates are (5,6). It will be multiplied, during different iterations, by the weight matrix elements $w_1(5,6)$, $w_2(5,3)$, $w_6(2,6)$, and $w_7(2,3)$. Consequently, one may choose these weight elements such they add up to unity. More generally stated, embodiment weight matrices may be chosen such that for any image pixel element p, the sum of the different weight elements multiplying p during compression are chosen to add up to a predetermined value, which without loss of generality may be chosen to be unity.

[0029] An embodiment image synthesis method may be described as follows. The scalar weights $\sigma_i(k), i=1,\cdots,n(k)$, the set $\{u_i(k), i=1,\cdots,n(k)\}$, and the set $\{v_i(k), i=1,\cdots,n(k)\}$ will be given by the compression algorithm for each index k where the index k will range over the vertices in the partitioning of the original image matrix M, or at least that portion of the original image matrix chosen for compression, storage, or communication. For each such k, form synthesized sub-block $\hat{m}_k$ as follows:

$$\hat{m}_k = \sum_{i=1}^{n(k)} \sigma_i(k) u_i(k) v_i'(k).$$

The synthesized image $\hat{M}$ is obtained by overlaying the sub-blocks $\hat{m}_k$ in the same relative positions as the sub-blocks m_k in the original image M, and numerically adding elements where there is overlap.

[0030] The above method is illustrated in the flow diagram of Fig. 4. In block **402**, the weighted vector outer product is formed for a set of scalar weights and associated vectors to provide the synthesized sub-block $\hat{m}_k$. After all synthesized sub-blocks have been accounted for, in step **404** the synthesized sub-blocks are overlaid and added to provide the synthesized image matrix $\hat{M}$.

[0031] In other embodiments, the operations performed in block **404** may be interleaved with operations in block **402** to provide sub-blocks of $\hat{M}$ in a pipelined fashion, where these sub-blocks need not be of the same dimension as the sub-blocks $\hat{m}_k$ An example is provided by Fig. 5. In block **502**, the notation $\oplus$ in the expression $\hat{M} \oplus \hat{m}_k$ indicates that elements of $\hat{m}_k$ are added to those elements in the matrix $\hat{M}$

having the same coordinates as the elements in M associated with the sub-block m_k. That is, block **502** performs in iterative fashion the overlay and addition of step **404**. Note that for block **502** of Fig. 5, a slight abuse of notation is made in that $\hat{M}$ denotes a running sum, and is not equal to the synthesized image matrix until all iterations have been performed.

[0032] Various sub-blocks of the synthesized image will be available at various iterations of block **502**, which may be provided by graphics hardware **112** to a frame buffer (not shown in Fig. 1) when available. For example, for the simplified image example of Fig. 2, when block **502** has been performed such that the index k values have included 1, 2, 3, 6, 7, 8, 11, 12, and 13, then all pixels in the synthesized image having the same pixel coordinates as the pixels in Fig. 2 within the square having corners at vertices 7, 8, 12, and 13 have been computed and are ready for the frame buffer.

[0033] The embodiments described herein for image synthesis involve forming weighted sums of vector outer products and matrix addition. These types of operations are well suited for processors tuned for such vector operations. These processors may be employed in high performance graphics hardware cards, such as graphics hardware **112**, or perhaps may be part of a general programmable processor, such as microprocessor **102** in a computer system.

[0034] Many other embodiments may be practiced without departing from the scope of the invention as claimed below. For example, various representations making use of weighted vector outer products other than the singular value decomposition may be employed for compressing an image matrix. Other embodiments may not perform sub-block weighting during compression, but may perform sub-block weighting during synthesis. For example, block **304** in Fig. 3 need not necessarily be performed, and instead, the sub-blocks $\hat{m}_k$ in blocks **402** or **502** may be weighted by the weight matrices w_i, so that $\hat{m}_k * w_k$ is overlaid and summed over the running index k. However, weighting during synthesis increases the computational burden placed on graphics hardware **112**. Note also that the summation and overlay need not necessarily be performed in any particular order, so that the index k need not necessarily be incremented in a sequential manner.

[0035] Furthermore, the compression and synthesis embodiments claimed below need not be limited to matrices representing images. The subject matter of the claims below may be applied to any matrix, no matter how it is physically derived.